

MySQL运维面试基础题目

反复练习，争取不看答案，不参照资料，均能熟练操作出来。

一. MySQL操作实战练习

1. 开启MySQL服务

```
/etc/init.d/mysqld start  
  
service mysqld start  
  
systemctl start mysqld
```

2. 检测端口是否运行

```
lsof -i :3306  
  
netstat -lntup |grep 3306
```

3. 为MySQL设置密码或者修改密码。

```
mysql -uroot -ppassword -e "set passowrd for root = password('password')"  
  
mysqladmin -uroot password "NEWPASSWORD"  
  
##修改密码  
mysqladmin -uroot password oldpassword "NEWPASSWORD"  
  
use mysql;  
  
update user set password = PASSWORD('newpassword') where user = 'root';flush  
privileges;  
  
## 5.7版本以上  
alter user 'root'@'localhost' identified by 'root'
```

4. 登陆MySQL数据库。

```
mysql -uroot -ppassword
```

5. 查看当前数据库的字符集

```
show create database DB_NAME;
```

6. 查看当前数据库版本

```
mysql -v
```

```
mysql -uroot -ppassowrd -e "use mysql;select version();"
```

```
mysqladmin version -uroot -p
```

7. 查看当前登录的用户。

```
select user();
```

8. 创建GBK字符集的数据库legolas, 并查看已建库完整语句

```
create database legolas DEFAULT CHARSET GBK COLLATE gbk_chinese_ci;
```

```
show create database legolas;
```

9. 创建用户legolas, 使之可以管理数据库legolas

```
grant all on legolas.* to 'legolas'@'localhost' identified by 'Com.123456';
```

##8.0以上版本

```
create user 'legolas'@'localhost' identified by 'Com.123456';
```

```
grant all on legolas.* to 'legolas'@'localhost';
```

10. 查看创建的用户legolas拥有哪些权限

```
show grants for 'legolas'@'localhost'\G
```

11. 查看当前数据库里有哪些用户

```
select user,host from mysql.user;
```

12. 进入legolas数据库

```
use legolas;
```

13. 创建—innoDB GBK表test, 字段id int(4)和name varchar(16)

```
create table test ( id int(4), name varchar(16) ) engine=innodb default  
charset=gbk;
```

14. 查看建表结构及表结构的SQL语句

```
desc test;
```

```
show create table test\G
```

15. 插入一条数据"1,legolas"

```
insert into test values (1,'legolas');
```

16. 再批量插入2行数据 “2, 精灵”, “3,legolasedu”

```
insert into test values('2','精灵'),('3','legolasedu');
```

17. 查询名字为legolas的记录

```
select * from test where name='legolas';
```

18. 把数据id等于1的名字legolas更改为king;

```
update test set name='king' where id=1;
```

19. 在字段name前插入age字段, 类型tinyint(2)

```
alter table test add age tinyint(2) after id;
```

20. 不退出数据库,完成备份legolas数据库

```
mysql> system mysqldump -uroot -pCom.123456 -B legolas > /root/legolas.sql;
```

21. 删除test表中的所有数据, 并查看

```
mysql> delete from test;  
  
mysql> select * from test;
```

22. 删除表test和legolas数据库并查看

```
mysql> drop table test;  
  
mysql> show tables;  
  
mysql> drop database legolas;  
  
mysql> show databases;
```

23. 不退出数据库恢复以上删除的数据

```
mysql> system mysql -uroot -pCom.123456 < /root/legolas.sql;  
  
##或者  
mysql> source /root/legolas.sql;
```

24. 把库表的GBK字符集修改为UTF8。

```
mysql> alter database legolas default character set utf8;  
  
mysql> alter table test default character set utf8;
```

25. 把id列设置为主键, 在Name字段上创建普通索引。

```
mysql> alter table test add primary key(id);

mysql> alter table test add index name(name);
```

26. 在字段name后插入手机号字段(shouji), 类型char(11)。

```
mysql> alter table test add shouji char(11) after name;

##默认就是在最后一列后面插入新列
mysql> alter table test add shouji char(11);
```

27. 所有字段上插入2条记录 (自行设定数据)

```
mysql> insert into test values('4','23','li','13700000001'),
('5','26','zhao','13710000001');
```

28. 在手机字段上对前8个字符创建普通索引。

```
mysql> alter table test add index sj(shouji(8));
```

29. 查看创建的索引及索引类型等信息。

```
mysql> show index from test\G

##或者
mysql> show create table test;

##或者
mysql> show keys from test\G
```

30. 删除Name, shouji列的索引。

```
mysql> alter table test drop index name;

mysql> alter table test drop index sj;
```

31. 对Name列的前6个字符以及手机列的前8个字符组建联合索引。

```
mysql> alter table test add index (name(6),shouji(8));
```

32. 查询手机号以135开头的, 名字为legolas的记录 (提前插入) 。

```
mysql> select * from test where shouji rlike '^137' and name='legolas';
```

33. 查询上述语句的执行计划 (是否使用联合索引等) 。

```
mysql> explain select * from test where shouji rlike '^147' and name='legolas'\G
```

34. 把test表的引擎改成MyISAM。

```
mysql> alter table test engine=MyISAM;
```

35. 收回legolas用户的select权限。

```
mysql> revoke select on legolas.* from 'legolas'@'localhost';
```

36. 删除legolas用户。

```
mysql> drop user 'legolas'@'localhost';
```

37. 删除legolas数据库。

```
mysql> drop database legolas;
```

38. 使用mysqldadmin关闭数据库。

```
# mysqladmin -uroot -pCom.123456 shutdown
```

39. MySQL密码丢了，请找回？

1. 修改主配置文件/etc/my.cnf，在[mysqld]章节下添加：skip-grant-tables
2. 重启mysql服务：systemctl restart mysqld
3. 无密码登录mysql服务：mysql -uroot
4. 将mysql.user表中的root用户的authentication_string字段值设置为空，想等于清空root密码

```
mysql> update mysql.user set authentication_string='' where user='root';  
mysql> flush privileges;
```

5. 修改主配置文件/etc/my.cnf，将[mysqld]章节下添加的skip-grant-tables删除掉或者注释掉，然后重启MySQL服务；
6. 由于root用户的authentication_string字段内容已经清空，这时候还是可以无密码登录；
7. 登录MySQL服务后，修改root用户密码：

```
alter user 'root'@'localhost' identified by 'new_password';
```
8. 退出MySQL服务，使用新密码登录即可。

二. MySQL运维基础知识面试题

争取全部口头用自己的语言描述出来

1. 请解释关系型数据库概念及主要特点？

- + 关系型数据库模型是把复杂的数据结构归结为简单的二元关系，对数据的操作都是建立一个或多个关系表格上；
- + 最大的特点就是二维的表格，通过SQL结构查询语句存取数据，保持数据一致性方面很强大

2. 请说出关系型数据库的典型产品、特点及应用场景？

1. **mysql** 互联网企业常用
2. **oracle** 大型传统企业应用软件
3. **sql server windows**平台使用

关系型数据库使用场景：

- + 需要做复杂处理数据；
- + 数据量不是特别大的数据；
- + 对安全性能要求高的数据；
- + 数据格式单一的数据；

3. 请解释非关系型数据库概念及主要特点？

NoSQL数据库的主要技术特点有以下几种：

- + 使用弱存储模式技术。简化了约束要求，提高处理速度
- + 没有采用SQL技术标准来定义和操作数据库
- + 采用弱事务保证数据可用性及安全性或根本没有事务处理机制
- + 主要采用多机分布式处理（DP,Distributed Processing）方式

NoSQL 技术主要解决以互联网业务应用为主的大数据应用问题，重点要突出处理速度的响应和海量数据的存储问题。特点：高性能、高并发、简单易安装。

4. 请说出非关系型数据库的典型产品、特点及应用场景？

1. **memcached、Redis** 键值数据库，存储在内存中，可以做缓存
主要用来缓存，数据库，消息中间件
2. **MongoDB** 面向文档的数据库
 - + 游戏场景，使用 **MongoDB** 存储游戏用户信息，用户的装备、积分等直接以内嵌文档的形式存储，方便查询、更新
 - + 物流场景，使用 **MongoDB** 存储订单信息，订单状态在运送过程中会不断更新，以 **MongoDB** 内嵌数组的形式来存储，一次查询就能将订单所有的变更读取出来。
 - + 社交场景，使用 **MongoDB** 存储存储用户信息，以及用户发表的朋友圈信息，通过地理位置索引实现附近的人、地点等功能
 - + 物联网场景，使用 **MongoDB** 存储所有接入的智能设备信息，以及设备汇报的日志信息，并对这些信息进行分析
 - + 视频直播，使用 **MongoDB** 存储用户信息、礼物信息等
3. **HBase** 列族数据库，适合用来存储各种类型的大规模的数据，可为用户提供实时的在线查询，同时也支持离线的应用。
 - + 搜索引擎应用
 - + 捕获增量数据
 - + 存储监控参数
 - + 存储用户交互数据
 - + 存储遥测数据
 - + 广告效果和点击流
4. **Neo4j** 图数据库，
 - + 融行业应用：反欺诈多维关联分析场景
 - + 社交网络图谱
 - + 企业关系图谱
 - + 订单快递查询业务

5. 请详细描述SQL语句分类及对应代表性关键字。

DDL 数据定义语言，用来定义数据库对象：库、表、列

+ 代表性关键字：**create alter drop**

DML 数据操作语言，用来定义数据库记录

+ 代表性关键字：**insert delete update**

DCL 数据控制语言，用来定义访问权限和安全级别

+ 代表性关键字：**grant revoke**

DQL 数据查询语言，用来查询记录数据

+ 代表性关键字：**select**

6. 请详细描述char(4)和varchar(4)的差别。

char长度是固定不可变的，varchar长度是可变的（在设定内）

同样写入cn字符，char类型对应的长度是4(cn+两个空格)，但varchar类型对应长度是2

7. 如何创建一个utf8字符集的数据库legolas?

```
create database legolas default character utf8 collate utf8_general_ci;
```

8. 如何授权legolas用户从172.16.1.0/24访问数据库。

```
grant all on *.* to 'legolas'@'172.16.1.%' identified by '123456';
```

9. 什么是MySQL多实例，如何配置MySQL多实例?

mysql多实例就是在同一台服务器上启用多个mysql服务，它们监听不同的端口，运行多个服务进程，它们相互独立，互不影响的对外提供服务，便于节约服务器资源与后期架构扩展

多实例的配置方法有两种：

1. 一个实例一个配置文件，不同端口
2. 同一配置文件(my.cnf)下配置不同实例，基于mysqld_multi工具

10. 如何加强MySQL安全，请给出可行的具体措施?

1. 删除数据库不使用的默认用户
2. 配置相应的权限（包括远程连接）
3. 不可在命令行界面下输入数据库的密码
4. 定期修改密码与加强密码的复杂度

11. MySQL root密码忘了如何找回?

1. 修改主配置文件/etc/my.cnf, 在[mysqld]章节下添加: skip-grant-tables
2. 重启mysql服务: systemctl restart mysqld
3. 无密码登录mysql服务: mysql -uroot
4. 将mysql.user表中的root用户的authentication_string字段值设置为空, 想等于清空root密码

```
mysql> update mysql.user set authentication_string='' where user='root';
mysql> flush privileges;
```

5. 修改主配置文件/etc/my.cnf, 将[mysqld]章节下添加的skip-grant-tables删除掉或者注释掉, 然后重启MySQL服务;
6. 由于root用户的authentication_string字段内容已经清空, 这时候还是可以无密码登录;
7. 登录MySQL服务后, 修改root用户密码:

```
alter user 'root'@'localhost' identified by 'new_password';
```
8. 退出MySQL服务, 使用新密码登录即可。

12. delete和truncate删除数据的区别?

- + 前者删除数据可以恢复, 它是逐条删除速度慢, 不重置自动增长列
- + 后者是物理删除, 不可恢复, 它是整体删除速度快, 重置自动增长列

13. MySQL Sleep线程过多如何解决?

1. 可以杀掉sleep进程, kill PID
 2. 修改配置, 重启服务
- ```
[mysqld]
wait_timeout = 600
interactive_timeout=30
```

#如果生产服务器不可随便重启可以使用下面的方法解决

```
set global wait_timeout=600
set global interactive_timeout=30;
```

## 14. sort\_buffer\_size参数作用? 如何在线修改生效?

- + 在每个connection(session)第一次连接时需要使用到, 来提高访问性能
- + set global sort\_buffer\_size = 2M

## 15. 如何在线正确清理MySQL binlog?

- + MySQL中的binlog日志记录了数据中的数据变动, 便于对数据的基于时间点和基于位置的恢复;
- + 但日志文件会越来越大, 占用大量的磁盘空间, 因此需要定时清理一部分日志信息

手工删除:

1. 首先查看主从库正在使用的binlog文件名称  

```
show master(slave) status\G
```
2. 删除之前一定要备份  

```
purge master logs before '2017-09-01 00:00:00'; ##删除指定时间前的日志
purge master logs to 'mysql-bin.000001'; ##删除指定的日志文件
```

自动删除:

- + 通过设置binlog的过期时间让系统自动删除日志  

```
show variables like 'expire_logs_days'; ##查看过期时间
set global expire_logs_days = 30; ##设置过期时间
```

## 16. binlog工作模式有哪些？各有什么特点，企业如何选择？

1. **STATEMENT**：记录实际的SQL语句
2. **ROW**：记录每行所做的更改。  
例如，更新语句更新10行，所有10行的更新信息都会被写入日志。  
而在基于语句的复制中，只有更新语句会被写入日志，默认格式是**ROW**。
3. **MIXED**：当需要时，MySQL会从**STATEMENT**切换到**ROW**。  
有些语句在不同服务器上执行时可能会得到不同的结果。  
例如，**UUID()**函数的输出就因服务器而异。  
这些语句被称为非确定性的语句，基于这些语句的复制是不安全的。  
在这些情况下，当设置**MIXED**格式时，MySQL服务器会切换为基于行的格式。

## 17. 误操作执行了一个drop库SQL语句，如何完整恢复？

1. 停止主从复制，在主库上执行锁表并刷新binlog操作，接着恢复之前的全备文件（比如0点的全备）
2. 将0点时的binlog文件与全备到故障期间的binlog文件合并导出成sql语句  
`mysqlbinlog --no-defaults mysql-bin.000011 mysql-bin.000012 > bin.sql`
3. 将导出的sql语句中drop语句删除，恢复到数据库中  
`mysql -uroot -pmysql123 < bin.sql`

## 18. mysqldump备份使用了-A -B参数，如何实现恢复单表？

- A 此参数作用是备份所有数据库（相当于--all-databases）
- B databasename 备份指定数据（单库备份使用）

### 方法1:

当要恢复的单个数据库仍存在于运行中的数据库时，可用下列方法恢复：

如该库已被删除，需先进入数据库中建立该库，再进行如下操作

示例：恢复test库

<1>从全备中直接导入单个库

格式：

`mysql -u用户 -p密码 单个数据库名 --one-database < 全备的sql文件`

命令：

`mysql -uroot -p'@#¥vse12' test -o < all.2018-12.sql`

备注：

-o 为 --one-database的缩写，也可使用 --one-database

### 方法2:

示例：恢复zabbix库

<1>从全备中直接导出zabbix库数据到新的sql文件

格式：

`sed -n '/^-- Current Database: `库名`/,/^-- Current Database: `/p' 全备sql文件 > 新sql文件`

命令：

`sed -n '/^-- Current Database: `zabbix`/,/^-- Current Database: `/p' all.2018-12.sql > zabbix.sql`

备注:

要注意空格, 注意符号

**Current Database:**后必有一个空格

<2>将提取出的新的sql数据导入数据库

格式:

`mysql -u用户 -p密码 < 单个库sql文件`

命令:

`mysql -uroot -p'@#¥vse12' < zabbix.sql`

## 19. 详述MySQL主从复制原理及配置主从的完整步骤。

主从复制原理:

1. **Slave**上面的**IO**线程连接上**Master**, 并请求从指定**Binary log**文件的指定位置(或者从最开始的日志)之后的日志内容;
2. **Master**接收到来自**Slave**的**IO**线程的请求后, 通过负责复制的**IO**线程根据请求信息读取指定日志指定位置之后的日志信息, 返回给**Slave**端的**IO**线程。  
返回信息中除了日志所包含的信息之外, 还包括本次返回的信息在**Master**端**Binary log**文件的名称以及在**Binary log**中的位置;
3. **Slave**的**IO**线程收到信息后, 将接收到的日志内容依次写入到**Slave**端的**RelayLog**文件(`mysql-relay-bin.xxxxx`)的最末端, 并将读取到的**Master**端的**bin-log**的文件名和位置记录到**master-info**文件中, 以便在下一次读取的时候能够清楚的告诉**master**“我需从某个**bin-log**的哪个位置开始往后的日志内容, 请发给我”;
4. **Slave**的**SQL**线程检测到**Relay Log**中新增加了内容后, 会马上解析该**Log**文件中的内容成为在**Master**端真实执行时候的那些可执行的查询或操作语句, 并在自身执行那些查询或操作语句, 这样, 实际上就是在**master**端和**Slave**端执行了同样的查询或操作语句, 所以两端的数据是完全一样的。

主从复制操作步骤:

1. 在主库上启用二进制日志记录
2. 在主库上创建一个复制用户
3. 在从库上设置唯一的**server\_id**
4. 从主库中取得备份
5. 恢复从库上的备份
6. 执行**CHANGE MASTER TO**命令
7. 开始复制

## 20. 如何开启从库的binlog功能?

1. 修改配置文件

`[mysqld]`

`log_bin=/data/mysql/binlogs/server1`

`server_id=100`

2. 重启MySQL服务

## 21. MySQL如何实现双向互为主从复制, 并说明应用场景?

双向同步主要应用于解决单一主库写的压力, 具体配置如下

主库配置

```
[mysqld]
```

```
auto_increment_increment = 2 #起始ID
```

```
auto_increment_offset = 1 #ID自增间隔
```

```
log-slave-updates
```

从库配置

```
[mysqld]
```

```
auto_increment_increment = 2 #起始ID
```

```
auto_increment_offset = 2 #ID自增间隔
```

```
log-slave-updates
```

主从库服务器都需要重启mysql服务

## 22. MySQL如何实现级联同步，并说明应用场景？

级联同步主要应用在从库需要做为其它数据库的主库

如果想在两个数据中心之间设置复制，一般会使用这种方式。

主库（**the primary master**）及其从库将位于一个数据中心内。

辅助主库（中继主库）从另一个数据中心的主库进行复制。

另一个数据中心的所有从库都从辅助主库（**the secondary master**）复制。

## 23. MySQL主从复制故障如何解决？

登陆从库

1. 执行`stop slave;`停止主从同步
2. 然后`set global sql_slave_skip_counter = 1;`跳过一步错误
3. 最后执行 `start slave;`并查看主从同步状态

需要重新进行主从同步操作步骤如下：

进入主库

1. 进行全备数据库并刷新binlog，查看主库此时的状态
2. 恢复全备文件到从库，然后执行`change master`
3. 开启主从同步`start slave;`并查看主从同步状态

## 24. 如何监控主从复制是否故障？

```
1. mysql -uroot -ppassowrd -e "show slave status\G" |grep -E
"Slave_IO_Running|Slave_SQL_Running"|awk '{print $2}'|grep -c Yes
```

2. 通过判断Yes的个数来监控主从复制状态，正常情况等于2

## 25. MySQL数据库如何实现读写分离？

1. 基于程序代码内部实现
2. 基于中间代理层实现：例如：Mycat、MySQL Router

## 26. 生产一主多从从库宕机，如何手工恢复？

1. 执行`stop slave` 或者停止服务
2. 修复好从库数据库
3. 然后重新操作主库同步

## 27. 生产一主多从主库宕机，如何手工恢复？

1. 登陆各个从库停止同步，并查看谁的数据最新，将它设置为新主库让其它从库同步其数据
2. 修复好主库之后，重新操作主从同步的步骤就可以了
  - + 需要注意新的主库如果之前是只读，需要关闭此功能让其可写
  - + 需要在新从库创建与之前主库相同的同步的用户与权限
  - + 其它从库执行`change master to master_port=新主库的端口, start slave`

## 28. 工作中遇到过哪些数据库故障，请描述2个例子？

1. 开发使用`root`用户在从库上写入数据造成主从数据不一致，并且前端没有展示需要修改的内容（仍旧是老数据）
2. 内网测试环境服务器突然断电造成主从同步故障

## 29. MySQL出现复制延迟有哪些原因？如何解决？

1. 需要同步的从库数据太多
2. 从库的硬件资源较差，需要提升
3. 网络问题，需要提升网络带宽
4. 主库的数据写入量较大，需要优配置和硬件资源
5. `sql`语句执行过长导致，需要优化

## 30. 给出企业生产大型MySQL集群架构可行备份方案？

1. 双主多从，主从同步的架构，然后实行某个从库专业做为备份服务器
2. 编写脚本实行分库分表进行备份，并加入定时任务
3. 最终将备份服务推送至内网专业服务器，数据库服务器本地保留一周
4. 备份服务器根据实际情况来保留备份数据（一般30天）

## 31. 什么是数据库事务，事务有哪些特性？企业如何选择？

事务就是一组应该一起成功或一起失败的SQL语句。

事务应该具备原子性、一致性、隔离性和持久性（Atomicity、Consistency、Isolation、Durability,简称ACID）的属性。

- + 原子性：所有的SQL语句要么全部成功，要么全部失败，不会存在部分更新。
- + 一致性：事务只能以允许的方式改变受其影响的数据。
- + 隔离性：同时发生的事务（并发事务）不应该导致数据库处于不一致的状态中。  
系统中每个事务都应该像唯一事务一样执行。任何事务都不应影响其他事务的存在。
- + 持久性：无论数据库或系统是否发生故障，数据都会永久保存在磁盘上，并且不会丢失。

## 32. 请解释全备、增备、冷备、热备概念及企业实践经验？

- + 全备：数据库所有数据的一次完整备份，也就是备份当前数据库的所有数据
- + 增备：就在上次备份的基础上备份到现在所有新增的数据
- + 冷备：停止服务的基础上进行备份操作
- + 热备：实行在线进行备份操作，不影响数据库的正常运行
- + 全备在企业中基本上是每周或天一次，其它时间是进行增量备份
- + 热备使用的情况是有两台数据库在同时提供服务的情况，针对归档模式的数据库
- + 冷备使用情况有企业初期，数据量不大且服务器数量不多，可能会执行某些库、表结构等重大操作时

### 33. MySQL的SQL语句如何优化？

建立主键与针对查询条件添加索引

### 34. 企业生产MySQL集群架构如何设计备份方案？

1. 集群架构可采用双主多从的模式，但实际双主只有一主在线提供服务，两台主之间做互备；
2. 另外的从可做读的负载均衡，然后将其中一台抽出专业做备份；

### 35. 开发有一堆数据发给dba执行，DBA执行需注意什么？

1. 需要注意语句是否有格式上的错误，执行出错会导致过程中断；
2. 还需要注意语句的执行时间是否过长，是否会对服务器负载产生压力，影响实际生产；

### 36. 如何调整生产线中MySQL数据库的字符集。

1. 首先导出库的表结构 `-d` 只导出表结构，然后批量替换
2. 导出库中的所有数据（在不产生新数据的前提下）
3. 然后全局替换 `set names = xxxxx`
4. 删除原有库与表，并新创建出来，再导入建库与建表语句与所有数据

### 37. 请描述MySQL里中文数据乱码原理，如何防止乱码？

服务器系统、数据库、客户端三方字符集不一致导致，需要统一字符

### 38. 企业生产MySQL如何优化（请多角度描述）？

1. 商业需求合理化
2. 系统架构及实现的影响
3. 插叙语句优化
4. 数据库 `schema` 设计的合理化
5. 服务器硬件及网络资源提升

### 39. MySQL高可用方案有哪些，各自特点，企业如何选择？

1. 主从或主主半同步复制
2. MHA或者MMM高可用
3. 基于共享存储方案
4. 基于磁盘复制方案
5. 分布式解决方案：MGR或者PXC

#### 40. 如何分表分库备份及批量恢复（口述脚本实现过程）？

实现思路：

1. 先取出MySQL所有库的名称
2. 用for循环遍历每个库
3. 再遍历每个库中的每个表
4. 然后通过mysqldump进行备份

```
#!/bin/bash
password='Com.123456'
for db_name in `mysql -u root -p${password} -e "show databases;" 2> /dev/null | sed 1d`
do

 mkdir -p /backup/${db_name}

 for table_name in `mysql -uroot -p${password} -e "show tables from ${db_name}" 2> /dev/null | sed 1d`
 do

 /usr/bin/mysqldump -uroot -p${password} ${db_name} ${table_name} >
 /backup/${db_name}/${date +%F}${table_name}.sql

 done

done
```

#### 41. 如何批量更改数据库表的引擎？

1. 查询数据库中表的名称
2. 使用：alter table 表名 engine='存储引擎' 命令进行修改

```
#!/bin/bash
DB=company
PASSWD='Com.123456'

for table_name in `mysql -uroot -p${PASSWD} -e "show tables from ${DB}" 2> /dev/null | sed 1d`
do
 mysql -uroot -p${PASSWD} -e "alter table ${DB}.${table_name} engine='MyISAM'" 2> /dev/null
done
```

#### 42. 如何批量更改数据库字符集？

43. 网站打开慢，请给出排查方法，如是数据库慢导致，如何排查并解决，请分析并举例？

44. xtrabackup的备份、增量备份及恢复的工作原理？

45. 误执行drop数据，如何通过xtranbackup恢复？
46. 如果做主从数据一致性校验？
47. 如果监控MySQL的增删改查次数？
48. MySQL索引的种类及工作原理。
49. 请描述MySQL不同引擎锁的机制。
50. 请描述InnoDB支持的四种事务隔离级别名称及特点。
51. 如何自定义脚本启动MySQL(说出关键命令)
52. 如何自定义脚本平滑关闭MySQL(说出关键命令)
53. MySQL MHA高可用软件的详细工作原理
54. 你们的公司如何实现数据库读写分离的？
55. 请简单描述下MySQL的体系结构知识？
56. 请简单描述下InnoDB引擎的内部结构原理？